

```
entry:
%stktop_4 = alloca i8, i32 48, align 1
%tos = ptrtoint ptr %stktop_4 to i64
%0 = add i64 %tos, 12
%RBP_N.36 = inttoptr i64 %0 to ptr
%1 = add i64 %tos, 16
%RBP_N.32 = inttoptr i64 %1 to ptr
%2 = add i64 %tos, 20
%RBP_N.28 = inttoptr i64 %2 to ptr
%3 = add i64 %tos, 24
%RBP_N.24 = inttoptr i64 %3 to ptr
%4 = add i64 %tos, 32
%RBP_N.16 = inttoptr i64 %4 to ptr
%5 = add i64 %tos, 40
%RBP_N.8 = inttoptr i64 %5 to ptr
%6 = add i64 %tos, 0
%RSP_P.0 = inttoptr i64 %6 to ptr
store i64 3735928559, ptr %RSP_P.0, align 8
%RBP = ptrtoint ptr %RSP_P.0 to i64
store i64 %arg1, ptr %RBP_N.8, align 1
store i64 %arg2, ptr %RBP_N.16, align 1
store i64 %arg3, ptr %RBP_N.24, align 1
store i32 %arg4, ptr %RBP_N.28, align 1
%memload = load i32, ptr getelementptr inbounds ([112 x i8], ptr @rodata_15,
... i32 0, i32 4), align 1, !ROData.Content !1
store i32 %memload, ptr %RBP_N.32, align 1
store i32 0, ptr %RBP_N.36, align 1
br label %bb.1
```

```
bb.1:
%memload1 = load i32, ptr %RBP_N.36, align 1
%7 = load i32, ptr %RBP_N.28, align 1
%8 = sub i32 %memload1, %7
%9 = call { i32, i1 } @llvm.usub.with.overflow.i32(i32 %memload1, i32 %7)
%CF = extractvalue { i32, i1 } %9, 1
%ZF = icmp eq i32 %8, 0
%highbit = and i32 -2147483648, %8
%SF = icmp ne i32 %highbit, 0
%10 = call { i32, i1 } @llvm.ssub.with.overflow.i32(i32 %memload1, i32 %7)
%OF = extractvalue { i32, i1 } %10, 1
%11 = and i32 %8, 255
%12 = call i32 @llvm.ctpop.i32(i32 %11)
%13 = and i32 %12, 1
%PF = icmp eq i32 %13, 0
%CmpSFOF_JGE = icmp eq i1 %SF, %OF
br i1 %CmpSFOF_JGE, label %bb.3, label %bb.2
```

|   |   |
|---|---|
| T | F |
|---|---|

```
bb.2:
%memload2 = load i64, ptr %RBP_N.16, align 1
%memload3 = load i64, ptr %RBP_N.36, align 1
%14 = trunc i64 %memload3 to i32
%RCX = sext i32 %14 to i64
%memref-basereg = add i64 %memload2, %RCX
%15 = inttoptr i64 %memref-basereg to ptr
%memload4 = load i32, ptr %15, align 1
%16 = trunc i32 %memload4 to i8
%EDX = sext i8 %16 to i32
%memload5 = load i64, ptr %RBP_N.24, align 1
%memload6 = load i64, ptr %RBP_N.36, align 1
%17 = trunc i64 %memload6 to i32
%RCX7 = sext i32 %17 to i64
%memref-basereg8 = add i64 %memload5, %RCX7
%18 = inttoptr i64 %memref-basereg8 to ptr
%memload9 = load i32, ptr %18, align 1
%19 = trunc i32 %memload9 to i8
%ESI = sext i8 %19 to i32
%EDX10 = mul nsw i32 %EDX, %ESI
%memload11 = load i64, ptr %RBP_N.8, align 1
%memload12 = load i64, ptr %RBP_N.36, align 1
%20 = trunc i64 %memload12 to i32
%RCX13 = sext i32 %20 to i64
%memref-basereg14 = add i64 %memload11, %RCX13
%21 = trunc i32 %EDX10 to i8
%22 = inttoptr i64 %memref-basereg14 to ptr
store i8 %21, ptr %22, align 1
%memload15 = load i32, ptr %RBP_N.36, align 1
%EAX = add i32 %memload15, 1
%23 = call { i32, i1 } @llvm.uadd.with.overflow.i32(i32 %memload15, i32 1)
%CF16 = extractvalue { i32, i1 } %23, 1
%24 = and i32 %EAX, 255
%25 = call i32 @llvm.ctpop.i32(i32 %24)
%26 = and i32 %25, 1
%PF17 = icmp eq i32 %26, 0
%ZF18 = icmp eq i32 %EAX, 0
%highbit19 = and i32 -2147483648, %EAX
%SF20 = icmp ne i32 %highbit19, 0
%27 = call { i32, i1 } @llvm.sadd.with.overflow.i32(i32 %memload15, i32 1)
%OF21 = extractvalue { i32, i1 } %27, 1
store i32 %EAX, ptr %RBP_N.36, align 1
br label %bb.1
```

```
bb.3:
store i32 0, ptr %stktop_4, align 1
br label %bb.4
```

```
bb.4:
%memload22 = load i32, ptr %stktop_4, align 1
%28 = load i32, ptr %RBP_N.28, align 1
%29 = sub i32 %memload22, %28
%30 = call { i32, i1 } @llvm.usub.with.overflow.i32(i32 %memload22, i32 %28)
%CF23 = extractvalue { i32, i1 } %30, 1
%ZF24 = icmp eq i32 %29, 0
%highbit25 = and i32 -2147483648, %29
%SF26 = icmp ne i32 %highbit25, 0
%31 = call { i32, i1 } @llvm.ssub.with.overflow.i32(i32 %memload22, i32 %28)
%OF27 = extractvalue { i32, i1 } %31, 1
%32 = and i32 %29, 255
%33 = call i32 @llvm.ctpop.i32(i32 %32)
%34 = and i32 %33, 1
%PF28 = icmp eq i32 %34, 0
%CmpSFOF_JGE49 = icmp eq i1 %SF26, %OF27
br i1 %CmpSFOF_JGE49, label %bb.6, label %bb.5
```

|   |   |
|---|---|
| T | F |
|---|---|

```
bb.5:
%memload29 = load i64, ptr %stktop_4, align 1
%35 = trunc i64 %memload29 to i32
%RAX = sext i32 %35 to i64
%36 = getelementptr i8, ptr %RBP_N.32, i64 %RAX
%memload30 = load i32, ptr %36, align 1
%37 = trunc i32 %memload30 to i8
%ECX = sext i8 %37 to i32
%memload31 = load i64, ptr %RBP_N.8, align 1
%memload32 = load i64, ptr %stktop_4, align 1
%38 = trunc i64 %memload32 to i32
%RDX = sext i32 %38 to i64
%memref-basereg33 = add i64 %memload31, %RDX
%39 = inttoptr i64 %memref-basereg33 to ptr
%memload34 = load i32, ptr %39, align 1
%40 = trunc i32 %memload34 to i8
%ESI35 = sext i8 %40 to i32
%ESI39 = add nsw i32 %ESI35, %ECX
%highbit36 = and i32 -2147483648, %ESI39
%SF37 = icmp ne i32 %highbit36, 0
%ZF38 = icmp eq i32 %ESI39, 0
%memref-basereg40 = add i64 %memload31, %RDX
%41 = trunc i32 %ESI39 to i8
%42 = inttoptr i64 %memref-basereg40 to ptr
store i8 %41, ptr %42, align 1
%memload41 = load i32, ptr %stktop_4, align 1
%EAX48 = add i32 %memload41, 1
%43 = call { i32, i1 } @llvm.uadd.with.overflow.i32(i32 %memload41, i32 1)
%CF42 = extractvalue { i32, i1 } %43, 1
%44 = and i32 %EAX48, 255
%45 = call i32 @llvm.ctpop.i32(i32 %44)
%46 = and i32 %45, 1
%PF43 = icmp eq i32 %46, 0
%ZF44 = icmp eq i32 %EAX48, 0
%highbit45 = and i32 -2147483648, %EAX48
%SF46 = icmp ne i32 %highbit45, 0
%47 = call { i32, i1 } @llvm.sadd.with.overflow.i32(i32 %memload41, i32 1)
%OF47 = extractvalue { i32, i1 } %47, 1
store i32 %EAX48, ptr %stktop_4, align 1
br label %bb.4
```

```
bb.6:
ret i32 %memload22
```